REMARKS

Applicants are in receipt of the Office Action mailed November 17, 2004. Claims 71-90 have been amended to recite a "computer accessible medium" instead of a "carrier medium". No claims have been added or canceled. Therefore, claims 1-90 remain pending in the application. Reconsideration is respectfully requested in light of the following remarks.

Section 102(e) Rejection:

The Office Action rejected claims 1, 9-11, 25, 27-28, 30-33, 34-39, 40, 41, 44, 46-49, 50, 51, 58-61, 62-66, 69, 70, 71, 75-77 and 84-90 were rejected under 35 U.S.C. § 102(e) as being anticipated by Tuatini (U.S. Publication 2002/0035645).

Regarding claim 1, contrary to the Examiner's assertion, Tuatini fails to disclose a compilation process of a virtual machine <u>converting a first computer programming language object into a data representation language representation of the first object</u>; wherein the data representation language representation of the first object is <u>configured for use in generating a copy</u> of the first object.

Tuatini teaches an application framework for developing applications including action and view handlers. Tuatini's action handlers implement business logic while view handlers control the formatting of the results returned by the business logic. Tuatini's application framework receives service requests from client computers and invokes appropriate action handlers to service the requests. The view handlers format responses and send responses to the requesting clients. (*see*, Tuatini, Abstract and paragraph 43). However, Tuatini fails to teach a compilation process of a virtual machine converting a first object into a data representation language representation of the first object, wherein the data representation language representation of the first object is <u>configured to use in generating a copy</u> of the first object.

The Examiner cites FIG. 2 and paragraph 79 of Tuatini and argues that Tuatini disclose that XML is convertible to and from JAVA objects. However, the cited passage describes generating JAVA objects to represent and manipulate the data in XML messages and regenerating the XML messages from the data in the JAVA objects. The XML data from request messages are not representations of the JAVA objects. Instead, Tuatini's Java objects allow for the programmatic manipulation of the data from the XML messages. Tuatini teaches that request messages are read in and translated from a client format to an application format and that responses are translated into the client format and written out as XML messages (page 3, paragraph 0049-0050). Tuatini teaches how his deserialize method is passed an indication of the client format [of the message], the message to be translated, and an indication of the application format. Tuatini's system then "deserializes the message and returns the JAVA object representing the message" and "performs any processing necessary to convert the message from the client format to the application format" (Tuatini, page 10, paragraph 0083-0084). Thus, any XML response message serialized from one of Tuatini's objects is not a data representation language *representation of a computer programming language object*, but instead is an XML response message in a client format different from the application format used by the object. Tuatini's system converts messages between formats to enable communication across platforms and between new and legacy applications.

Furthermore, in Tuatini an XML response message in a client format converted from a JAVA object in an application format is not a data representation language representation of the JAVA object. Nor is it configured for use in generating a copy of the JAVA object. Tuatini teaches that the data in the XML message is in a different format from the data in the JAVA object and thus is not configured to use is generating a copy of the object. Tuatini does not teach a data representation language representation of a first computer programming language object wherein the data representation language representation is configured for use in generating a copy of the first object

Applicants assert that the section 102 rejection of claim 1 is not supported by the cited prior art because a rejection under section 102 requires that the identical invention

must be shown in as complete detail as is contained in the claims and also requires the presence in a single prior art reference disclosure of <u>each and every element</u> of the claimed invention, <u>arranged as in the claim</u>. (M.P.E.P. § 2131). See also *Lindemann Maschinenfabrik GmbH v. American Hoist & Derrick Co.*, 221 USPQ 481, 485 (Fed. Cir. 1984).

Thus, for at least the reasons given above, the rejection of claim 1 is not supported by the prior art and removal thereof is respectfully requested. Remarks similar to those above regarding claim 1 also apply to claims 25, 40, 62, 71 and 84.

In further regard to claim 25, Tuatini additionally fails to disclose generating a message in the data representation language, wherein the message <u>includes the data representation language representation of a computer programming language object</u>; sending the message to a second process; and the second process <u>generating a copy of the computer programming language object</u> from the data representation language representation of the object included in the message.

The Examiner has not cited any portion of Tuatini in the rejection of claim 25. Instead, the Examiner appears to have inadvertently left a citation from the previous Office Action where the Examiner cited column 9, lines 24-42 and Figure 1 of Meltzer (USPAT 6,542,912). The Examiner's current rejection of claim 25 cites column 9, lines 24-42 and FIG. 1 of Tuatini. The current Tuatini citation is clearly incorrect as the Tuatini reference does not include column and line number, but instead uses paragraph numbering. Therefore, no *prima facie* rejection has been stated form claim 25.

Furthermore, as described above regarding claim 1, Tuatini does not disclose generating a data representation language representation of a computer programming language object. Instead, Tuatini converts XML messages from a client format to an application format. Thus, the XML messages in Tuatini are not and do not include representations of computer programming language objects.

Furthermore, Tuatini does not disclose generating a message in the data representation language, <u>wherein the message includes the data representation language representation of the computer programming language object</u> and sending the message to a second process. Instead, Tuatini teaches that the server receives a request message in XML from a client; translates the XML data from a client format into an application format using a JAVA object; generates results from the request; generates a response message in XML formatted according to the client format; and sends the response message back to the client (Tuatini, page 2, paragraph 0043 and page 3, paragraphs 0049-0050). The XML based request messages sent from clients to servers in Tuatini's system are request messages that do not include data representation language representations of the computer programming language objects. Just because Tuatini teaches creating a JAVA object from a translated XML request message does not mean that the request message was a representation of the object. In other words, rather than being representation of programming objects, Tuatini's XML request messages are simply requests for services.

Tuatini also fails to disclose the second process <u>generating a copy of the computer programming language object</u> from the data representation language representation of the object included in the message. Instead, as noted above, Tuatini only teaches generating JAVA objects that represent converted or translated versions of client request messages. Tuatini does not teach that client send representations of computer programming language objects, but instead only teaches that client send requests for services.

For at least the reasons given above, the rejection of claim 25 is not supported by the prior art and its removal is respectfully requested. Remarks similar to those above regarding claim 25 also apply to claims 50, 62 and 84.

Regarding claim 34, the Examiner has rejected claim 34 for the same corresponding reasons put forth in the rejection of claim 25 stating, "[i]t has the same inter process exchange of information." However, as noted above, no proper rejection was stated for claim 25. Furthermore, Applicants note that claims 25 and 34 are very

different in scope. The Examiner's rejection of claim 34 is improper since it is based on the same arguments as claim 25 even though claims 25 and 34 contain different limitations.

Applicants further note that Tuatini does not anticipate Applicants' claim 34. Specifically, Tuatini fails to teach a virtual machine generating an object from information representing the object. As described above regarding claim 1, Tuatini fails to disclose the generating a computer programming language object from a *representation of that object*. Firstly, Tuatini teaches that JAVA objects are generated from translated XML request messages (paragraphs 0049-50 and 0081-0083), not from representations of computer programming language objects. Just because Tuatini teaches creating an object from a translated XML request message does not mean that the request message was a representation of the object. In other words, rather than being representation of programming objects, Tuatini's XML request messages are simply requests for services. Generating JAVA objects to hold the data represented by the XML request messages does not constitute generating a computer programming language object from a data representation language representation of the object.

Thus, in light of the above remarks, Applicants assert that the rejection of claim 34 is not supported by the cited art and withdrawal of the rejection is respectfully requested. Similar remarks also apply to claim 50.

Section 103(a) Rejection:

The Office Action rejected claims 2, 26, 29, 52, 67 and 68 under 35 U.S.C. § 103(a) as being unpatentable over Tuatini in view of Mitchell (U.S. Patent 6,628,304). Applicants respectfully traverse this rejection for at least the reasons given above in regard to Tuatini.

Furthermore, regarding claim 2, contrary to the Examiner's contention, Tuatini in view of Mitchell does not teach or suggest wherein the first object references one or more

computer programming language objects, and wherein the compilation process of the virtual machine converting the first object into a data representation language representation of the first object comprises the compilation process converting the one or more objects into data representation language representations of the one or more objects.

Mitchell teaches a method for presenting hierarchical data to a user via a graphical user interface. Mitchell teaches that hierarchical data related to a computer network can be displayed using connected nodes arranged such that the nodes do not spatially interfere with other nodes. (See, Mitchell, Abstract and column 5, lines 18-36). Mitchell is concerned with graphically displaying hierarchical data and has nothing to do with converting computer programming objects into data representation language representations of the objects. The Examiner argues, "Mitchell discloses one object representing a hierarchy of multiple other objects" and cites column 5, lines 38-55 of Mitchell. However, the cited passage does not refer to computer programming objects. Instead, the cited passage of Mitchell describes how data representing data links in a computer network may be graphically displayed as a hierarchy of graphical user interface elements.

Mitchell's graphical objects have absolutely no bearing on one object referencing one or more computer programming language objects. A hierarchy of interconnected graphical representations of data does not imply anything regarding a computer programming language object referencing other computer programming objects.

Additionally, as noted above regarding the rejection of claim 1, Tuatini fails to teach converting a first object into a data representation language representation of the first object. As Mitchell has nothing to do with converting objects into data representation language representations of the objects, the Examiner's proposed combination of Tuatini and Mitchell also fails to teach or suggest converting one or more computer programming language objects into data representation language representations of the one or more computer programming language objects.

Thus, the Examiner's proposed combination of Tuatini and Mitchell does not result in a system that includes wherein a first object references one or more computer programming language objects and wherein converting the first object into a data representation language representation of the first object comprises converting the one or more objects into data representation language representations of the one or more computer programming objects. Instead, a combination of Tuatini and Mitchell results only in a system that generates JAVA objects to represent XML messages for service requests, as taught by Tuatini, and that displays hierarchical data, such as nodes in a computer network, in a graphical user interface, as taught by Mitchell.

Thus, for at least the reasons above, the rejection of claim 2 is not supported by the prior art and removal thereof is respectfully requested. Remarks similar to those above regarding claim 2 also apply to claims 26, 29, 52, and 67.

The Office Action rejected claims 3-7, 42-44, 53-55 and 72-74 under 35 U.S.C. § 103(a) as being unpatentable over Tuatini in view Humpleman et al. (U.S. Patent 6,546,419) (hereinafter "Humpleman"). Applicants respectfully traverse this rejection for at least the reasons given above in regard to Tuatini.

Furthermore, regarding claim 3, Tuatini in view of Humpleman does not teach or suggest processing the first object <u>into an intermediary table representation of the first object</u> and <u>processing the intermediary table representation</u> of the first object <u>into the data representation language representation</u> of the first object. The Examiner cites column 12, lines 55-57 where Humpleman describes how XML method messages are converted into an API format via a look-up table defining run-time translation if XML object method calls to device native language calls. However, using a look-up table to translate XML method definitions into native language calls does not imply processing the first object into an intermediary table representation of the first object. The look-up table in Humpleman is not a table representation of an object, but instead, includes translations for multiple XML method call definitions to corresponding native language calls.

Furthermore, Humpleman uses his look-up table for converting method calls definitions from XML to native API calls. Nowhere does Humpleman, or Tuatini, teach an intermediary table representation *of a computer programming language object.* The method call definitions in Humpleman are clearly not representations of computer programming language objects. Thus, nowhere does Tuatini or Humpleman, either singly or in combination, teach or suggest processing the first object into an intermediary table representation of the first object or processing the intermediary table representation of the first object into the data representation language representation of the first object.

For at least the reasons given above, the rejection of claim 3 is not supported by the prior art and its removal is respectfully requested.

Regarding claim 4, Tuatini in view of Humpleman fails to teach or suggest for each of one or more instance variables in the first object, generating an entry in the intermediary table representation of the first object, wherein the entry for each of the one or more instance variables includes an identifier of the instance variable and a value of the instance variable.

The Examiner cites column 13, lines 9-17 of Humpleman that describes providing a database of objects for making XML form method calls to remote API services. The cited passage has no relevance whatsoever to generating an entry in an intermediary table representation of an object, wherein the entry includes an identifier of an instance variable and a value of the instance variable. The Examiner argues that Humpleman's look-up table is "created at compile time for the purpose of converting XML messages to C language and vice versa, with class variables that are inherently identified with a value used to generate the intermediary table." However, the Examiner has failed to consider that since Humpleman's look-up table is created at compile time, it cannot possibly include values of instance variables which are inherently only available at run-time. Furthermore, as noted above, Humpleman's look-up table does not store variable values, but instead, provides translation definitions for translating XML method definitions into native language calls.

Neither Tuatini nor Humpleman, either separately or in combination, teach or suggest generating an entry in the intermediary table representation of the first object, wherein the entry for each of the one or more instance variables includes an identifier of the instance variable and a value of the instance variable. Thus, the rejection of claim 4 is not supported by the cited art and removal thereof is respectfully requested. Similar remarks as those above regarding claim 4 also apply to claims 43 and 55.

The Office Action rejected claims 12, 20, 22-24, 78 and 81-83 under 35 U.S.C. § 103(a) as being unpatentable over Tuatini in view of Ansari et al. (U.S. Patent 5,881,290) (hereinafter "Ansari"). Applicants respectfully traverse this rejection for at least the reasons given above in regard to Tuatini.

Furthermore, regarding claim 12, contrary to the Examiner's assertion, Tuatini in view of Ansari fails to teach or suggest a virtual machine receiving a data representation language representation of a first computer programming language object from a first process. The Examiner cites paragraph 0079 of Tuatini and argues that Tuatini discloses a method for generating JAVA object from XML. However, Tuatini does not generate JAVA objects from data representation language representations of computer programming language objects. Instead, Tuatini teaches generating JAVA object "through which attributes of the [request] message can be retrieved" (Tuatini, paragraph 0049). As discussed above regarding the 102(e) rejection of claim 25, Tuatini's request message are not, nor do they include, data representation language representations of computer programming language objects. Instead, Tuatini's request messages are requests for services sent from clients to servers. When processing received request message, Tuatini's server may convert the request from a client format to an application format and generate a JAVA object to load and access the data in the request message.

Tuatini in view of Ansari further fails to teach or suggest a decompilation process of the virtual machine generating the first object from the data representation language representation of the first object. The Examiner relies upon Ansari to teach a "method of

compiling and decompiling so that the original program information can undergo some processing" and cites column 3, lines 44-47 and column 9, lines 34-38 of Ansari.

Ansari teaches a compiler and a decompiler for industrial controllers that allow new instructions to be added to those already recognized by modifying an internal instruction table of instructions. (Ansari, column 2, lines 10-17). Applicants fail to see the relevance of Ansari's method for adding new instructions to an industrial controller compiler to Tuatini's application framework for client server communication.

The Examiner's cited references in Ansari (column 3, lines 44-47, and column 9, lines 34-38) refer to decompiling an industrial controller program into editable source. Applicant can find no reference in Ansari referring to generating an computer programming language object from a data representation language representation of the object, as the Examiner contends. The Examiner seems to be relying upon Ansari merely because Ansari teaches a "decompiler." The Examiner even states this directly, "Tuatini... does not disclose the translation (of XML into JAVA object) as a *decompilation* process ... [h]owever, Ansari explicitly discloses a method of compiling and decompiling." (emphasis added).

Applicants assert that Ansari's decompiler, which generates industrial controller source code from a program, does not teach generating an object from a data representation language representation of the object. The Examiner even supports this assertion by stating, "the combination [of Tuatini and Ansari] provides a means for the Tuatini invention to extract the object information so that it can be processed *as taught [by] Ansari*" (emphasis added). Applicants submit that the Examiner has failed to show how the combination of Tuatini and Ansari includes a virtual machine receiving a data representation language representation of a computer programming language object or that teaches or suggests a process of the virtual machine generating the object from the data representation language representation of the object.

Nowhere does Tuatini or Ansari, separately or in combination, teach or suggest a virtual machine receiving a data representation language <u>representation of a an object</u> or that teaches or suggests a process of the virtual machine generating the object from the data representation language representation of the object.

Thus, in light of the above remarks, applicants assert that the rejection of claim 12 is not supported by the cited art and withdrawal of the rejection is respectfully requested. Similar remarks as discussed above in regard to claim 12 apply to claim 78.

Regarding claim 20, Tuatini in view of Ansari fails to teach or suggest wherein the data representation language representation of the first object comprises <u>an identifier of the class of the first object</u>, contrary to the Examiner's assertion. The Examiner cites paragraph 79 of Tuatini and column 3, lines 44-47 and column 9, lines 34-38 of Ansari. However, as noted above, Tuatini teaches generating JAVA objects to manipulate translated versions of XML service request messages. Tuatini's request messages are not data representation language representations of objects and do not include an identifier of the class of an object. Specifically, Tuatini's system includes initializing the serialization service, which is responsible for generating JAVA objects for XML data, with XML to Java mappings (Tuatini, paragraph 0081-0082). Clearly, if Tuatini's XML request messages included identifiers of a JAVA class, there would be no need to load and use such XML to Java mappings.

Furthermore, since Ansari, as noted above, is silent regarding data representation language representations of objects, Ansari does not overcome any of Tuatini's deficiencies regarding a data representation language representation of an object including an identifier of the class of the object. Thus, the combination of Tuatini and Ansari clearly fails to teach or suggest wherein the data representation language representation of the first object comprises an identifier of the class of the first object, as erroneously asserted by the Examiner. The rejection of claim 20 is not supported by the prior art and removal thereof is respectfully requested.

The Office Action rejected claims 8, 45 and 57 under 35 U.S.C. § 103(a) as being unpatentable over Tuatini in view of Daly et al. (U.S. Patent 5,748,896) (hereinafter "Daly"). The Office Action rejected claims 13 and 14 under 35 U.S.C. § 103(a) as being unpatentable over Tuatini in view of Ansari, and further in view of Mitchell. The Office Action rejected claims 15-19, 56, 79 and 80 under 35 U.S.C. § 103(a) as being unpatentable over Tuatini in view of Ansari, and further in view of Humpleman. The Office Action rejected claim 21 under 35 U.S.C. § 103(a) as being unpatentable over Tuatini in view of Ansari, and further in view of Daly. Applicants respectfully traverse these rejections for at least the reasons given above in regard to Tuatini.

Applicants also assert that numerous ones of the dependent claims recite further distinctions over the cited art. However, since the independent claims have been shown to be patentably distinct, a further discussion of the dependent claims is not necessary at this time. Applicants reserve the right to present additional arguments at a later date if necessary.

Information Disclosure Statement:

Applicants note that three additional information disclosure statements with accompanying Forms PTO-1449 were submitted on July 19, 2001, September 17, 2001 and November 18, 2002, respectively. For the Examiner's convenience, Applicants have provided herewith copies of both the original form PTO-1449 and the date stamped return postcard from the three information disclosure statements referenced above **showing that these information disclosure statements and references were received by the USPTO.** Applicants request the Examiner to carefully consider the listed references and return copies of the signed and initialed Forms PTO-1449 from all three statements.
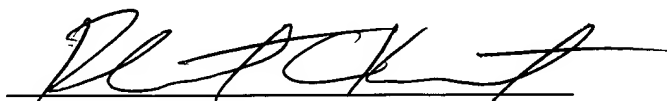
CONCLUSION

Applicants submit the application is in condition for allowance, and notice to that effect is respectfully requested.

If any extension of time (under 37 C.F.R. § 1.136) is necessary to prevent the above referenced application from becoming abandoned, Applicants hereby petition for such extension. If any fees are due, the Commissioner is authorized to charge said fees to Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C. Deposit Account No. 501505/5181-72000/RCK.

Also enclosed herewith are the following items:

☒ Return Receipt Postcard

☐ Petition for Extension of Time

☒ Copies of three previously submitted PTO-1449 forms and respective date stamped return postcards.

☐ Fee Authorization Form authorizing a deposit account debit in the amount of $ for fees (   ).

Respectfully submitted,

Robert C. Kowert
Reg. No. 39,255
ATTORNEY FOR APPLICANT(S)

Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8850

Date: ___February 17, 2005_____